**Exercise 3: Stored Procedures**

**Scenario 1:**The bank needs to process monthly interest for all savings accounts.

* + **Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

**Procedure:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

UPDATE Accounts

SET Balance = Balance \* 1.01

WHERE AccountType = 'Savings';

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Monthly interest applied to all savings accounts.');

END ProcessMonthlyInterest;

EXEC ProcessMonthlyInterest;

**Before Procedure call:**

1 1 Savings 450 06-AUG-24

2 2 Checking 2050 06-AUG-24

3 3 Savings 2000 06-AUG-24

4 4 Checking 2500 06-AUG-24

5 5 Savings 3000 06-AUG-24

6 6 Checking 3500 06-AUG-24

7 7 Savings 4000 06-AUG-24

8 8 Checking 4500 06-AUG-24

9 9 Savings 5000 06-AUG-24

10 10 Checking 5500 06-AUG-24

**After Procedure call:**

1 1 Savings 454.5 06-AUG-24

2 2 Checking 2050 06-AUG-24

3 3 Savings 2020 06-AUG-24

4 4 Checking 2500 06-AUG-24

5 5 Savings 3030 06-AUG-24

6 6 Checking 3500 06-AUG-24

7 7 Savings 4040 06-AUG-24

8 8 Checking 4500 06-AUG-24

9 9 Savings 5050 06-AUG-24

10 10 Checking 5500 06-AUG-24

**Scenario 2:**The bank wants to implement a bonus scheme for employees based on their performance.

* + **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

**Procedure:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

p\_department IN VARCHAR2,

p\_bonus\_percentage IN NUMBER

) AS

BEGIN

UPDATE Employees

SET Salary = Salary \* (1 + p\_bonus\_percentage / 100)

WHERE Department = p\_department;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to all employees in department: ' || p\_department);

END UpdateEmployeeBonus;

BEGIN

UpdateEmployeeBonus('IT',40);

END;

/

**Before Procedure call:**

1 Alice Johnson Manager 105000 HR 15-JUN-15

2 Bob Brown Developer 60000 IT 20-MAR-17

3 Charlie Wilson Analyst 55000 Finance 25-APR-18

4 Daisy Martinez Accountant 62000 Finance 30-AUG-16

5 Edward Clark Sales Manager 75000 Sales 10-JAN-19

6 Fiona Lewis HR Specialist 50000 HR 15-MAY-20

7 George Adams Developer 65000 IT 22-NOV-17

8 Hannah Scott Designer 56000 Marketing 05-SEP-19

9 Isaac Walker Developer 67000 IT 14-FEB-18

10 Julia King Marketing Manager 105000 Marketing 01-JUL-16

**After Procedure call:**

1 Alice Johnson Manager 105000 HR 15-JUN-15

2 Bob Brown Developer 117600 IT 20-MAR-17

3 Charlie Wilson Analyst 55000 Finance 25-APR-18

4 Daisy Martinez Accountant 62000 Finance 30-AUG-16

5 Edward Clark Sales Manager 75000 Sales 10-JAN-19

6 Fiona Lewis HR Specialist 50000 HR 15-MAY-20

7 George Adams Developer 127400 IT 22-NOV-17

8 Hannah Scott Designer 56000 Marketing 05-SEP-19

9 Isaac Walker Developer 131320 IT 14-FEB-18

10 Julia King Marketing Manager 105000 Marketing 01-JUL-16

**Scenario 3:**Customers should be able to transfer funds between their accounts.

* + **Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

**Procedure:**

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_from\_account IN NUMBER,

p\_to\_account IN NUMBER,

p\_amount IN NUMBER

) AS

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance

FROM Accounts

WHERE AccountID = p\_from\_account;

IF v\_balance<p\_amount THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient funds in account ' || p\_from\_account);

END IF;

BEGIN

UPDATE Accounts

SET Balance = Balance - p\_amount

WHERE AccountID = p\_from\_account;

UPDATE Accounts

SET Balance = Balance + p\_amount

WHERE AccountID = p\_to\_account;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transfer of ' || p\_amount || ' from account ' || p\_from\_account || ' to account ' || p\_to\_account || ' completed successfully.');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END;

END TransferFunds;

EXEC TransferFunds(2,3,50.0);

**Output:**

Procedure TRANSFERFUNDS compiled

Transfer of 50 from account 2 to account 3 completed successfully.

PL/SQL procedure successfully completed.